

Conference Paper Presentation
*History Matters: Incremental
Ontology Reasoning Using Modules*

Bernardo Cuenca Grau, Christian Halascheck-Wiener, and Yevgeny Kazakov

Joshua Taylor
September 18, 2008
Advanced Semantic Web
Winslow 4th Floor
Rensselaer Polytechnic Institute
Troy, NY 12180 USA

History Matters: Incremental Ontology Reasoning Using Modules

Bernardo Cuenca Grau¹, Christian Halaschek-Wiener²,
and Yevgeny Kazakov¹

¹The University of Manchester, School of Computer Science, Manchester M13 9PL, UK

²Department of Computer Science, University of Maryland, College Park, MD 20740, USA

ISWC 2007

Ontology Construction

- Lots of work goes into the initial version, and *everything* is new.
- Subsequent versions are produced by small, incremental changes.
- Many propositions entailed by a earlier version of an ontology are still entailed (by the same axioms) by a later version of the ontology.

- Since most of the axioms are the same, most of the inferences that were valid under the earlier version are valid in the later version.
- Current reasoners do not exploit this fact, but re-infer *all* consequences of the ontology.
- This prevents ontology developers from running reasoners as often as they could.

The phrase “all consequences” here refers to the classification of the ontology, that is, computing the subsumption relation for the concepts named by the ontology. Reclassifying the ontology is expensive, and this expense prevents developers from classifying as often as they might like. Unfortunately, this means that errors are not caught as soon as possible, and it makes interactive development impossible. (Cf. the comparison of interpreted/compiled-on-the-fly languages and those with edit-compile-test cycles.)

SHOIQ

- Signature: $\mathbf{S} = \mathbf{R} \cup \mathbf{C} \cup \mathbf{I}$
- SHOIQ-Role: $R \in \mathbf{R}$ or R^- where $R \in \mathbf{R}$.
- \mathbf{Rol} is the set of SHOIQ-roles for \mathbf{S} .
- $\mathbf{Con} ::= \perp \mid a \mid \neg C \mid C_1 \sqcap C_2 \mid \exists R.C \mid \geq n S.C$

It's hard to find a reference that says that OWL DL is SHOIQ. In fact, it's more common to see SHOIN(D). E.g., in "Pellet: A Practical OWL-DL Reasoner," the authors state that they have built "the first reasoner to support all of OWL-DL, i.e., the Description Logic (DL) SHOIN(D)" and also mention that OWL 1.1 is SROIQ(D). Nonetheless, in this paper they use SHOIQ as they describe it here. (It's worth noting Bernardo Cuenca Grau is a common author between that paper and this one.)

- An ontology is a set of *role inclusion axioms* ($R_1 \sqsubseteq R_2$), *transitivity axioms* ($\text{Trans}(R)$), and *general inclusion axioms* ($C_1 \sqsubseteq C_2$).
- The *signature* of an axiom α is the union of the role names, concept names, and nominals appearing in α .
- The signature of an ontology is the union of the signatures of the axioms of the ontology.

The notion of signature applies to arbitrary propositions in the system, not just axioms of the ontology. By defining the signature of an ontology as a function of the ontology, there is no need to /extend/ a signature in order to add an axiom. (It is not necessary to add a concept name before adding an axiom that mentions the concept.)

O^1 — Original Ontology

D1	$\text{CysticFibrosis} \equiv \text{Fibrosis} \sqcap \exists \text{locatedIn.Pancreas}$
D2	$\text{GeneticFibrosis} \equiv \text{Fibrosis} \sqcap \exists \text{hasOrigin.Genetic}$
D3	$\text{PancreaticFibrosis} \equiv \text{Fibrosis} \sqcap \text{PancreaticDisorder}$
C1	$\text{GeneticFibrosis} \sqsubseteq \text{GeneticDisorder}$
C2	$\text{PancreaticDisorder} \sqsubseteq \text{Disorder} \sqcap \exists \text{locatedIn.Pancreas}$

Wednesday, September 17, 2008

7

definitions:

D1 — Cystic Fibrosis is a Fibrosis which has at least one location being a Pancreas

D2 — Genetic Fibrosis is a Fibrosis that has at least one origin which is Genetic.

D3 — Anything which is a Pancreatic Disorder /and/ Fibrosis is Pancreatic Fibrosis.

concept inclusion axioms:

C1 — Every Genetic Fibrosis is a Genetic Disorder

C2 — Every Pancreatic Disorder is both a Disorder and has at least one location being a Pancreas.

O^2 — Modified Ontology

D1	$\text{CysticFibrosis} \equiv \text{Fibrosis} \sqcap \exists \text{locatedIn.Pancreas}$ $\sqcap \exists \text{hasOrigin.GeneticOrigin}$
D2	$\text{GeneticFibrosis} \equiv \text{Fibrosis} \sqcap \exists \text{hasOrigin.Genetic}$
D3	$\text{PancreaticFibrosis} \equiv \text{Fibrosis} \sqcap \text{PancreaticDisorder}$
C1	$\text{GeneticFibrosis} \sqsubseteq \text{GeneticDisorder}$
C2	$\text{PancreaticDisorder} \sqsubseteq \text{Disorder} \sqcap$ $\exists \text{locatedIn.Pancreas}$

The definition of Cystic Fibrosis is updated. Cystic Fibrosis is now understood to be the class of things which are Fibrosis, and have at least one location being a Pancreas, and at least one origin being a Genetic Origin.

α	Axiom	$O^1 \models \alpha?$	follows from	$O^2 \models \alpha?$	follows from
α_1	PancreaticFibrosis \sqsubseteq CysticFibrosis	Yes	D3,C2,D1	No	
α_2	CysticFibrosis \sqsubseteq GeneticDisorder	No		Yes	D1,D2,C1
α_3	PancreaticFibrosis \sqsubseteq Disorder	Yes	D3,C2	Yes	D3,C2
α_4	GeneticFibrosis \sqsubseteq CysticFibrosis	No		No	

(These probably oughtn't be called "axioms," but rather propositions.) The truth of "axioms" 3 and 4 don't change. 3 is easily seen. D3 and C2 remain unchanged, so their proof of axiom 3 is still valid (though, in some logical systems, e.g., FOL, an inconsistency introduced elsewhere could be used to make a valid proof of the negation of axiom 3). How to tell whether the truth of "axiom" 4 changes is more difficult.

Propositional Example

KB	
1	P
2	$P \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$

This is an example that gives the intuitive notion of module. $\{1,2,3\}$ is a module for R in the KB since the KB entails R , and the set $\{1,2,3\}$ a subset of KB also entails R . Similarly, $\{4\}$ is also a module for R in the KB.

Propositional Example

KB	
1	P
2	$P \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$

- $KB \models R?$

This is an example that gives the intuitive notion of module. $\{1,2,3\}$ is a module for R in the KB since the KB entails R , and the set $\{1,2,3\}$ a subset of KB also entails R . Similarly, $\{4\}$ is also a module for R in the KB.

Propositional Example

KB	
1	P
2	$P \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$

- $KB \models R$?
- How?

This is an example that gives the intuitive notion of module. $\{1,2,3\}$ is a module for R in the KB since the KB entails R , and the set $\{1,2,3\}$ a subset of KB also entails R . Similarly, $\{4\}$ is also a module for R in the KB.

Propositional Example

KB	
1	P
2	$P \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$

- $KB \models R$?
- How?
 - $\{1,2,3\} \models R$

This is an example that gives the intuitive notion of module. $\{1,2,3\}$ is a module for R in the KB since the KB entails R , and the set $\{1,2,3\}$ a subset of KB also entails R . Similarly, $\{4\}$ is also a module for R in the KB.

Propositional Example

KB	
1	P
2	$P \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$

- $KB \models R$?
- How?
 - $\{1,2,3\} \models R$
 - $\{4\} \models R$

This is an example that gives the intuitive notion of module. $\{1,2,3\}$ is a module for R in the KB since the KB entails R , and the set $\{1,2,3\}$ a subset of KB also entails R . Similarly, $\{4\}$ is also a module for R in the KB.

Modules

If an ontology entails an axiom, then a module for the axiom must contain enough for at least one proof of that axiom. A module for a signature contains enough information to decide whether the ontology entails an axiom constructed from the signature. “Hence, knowing all the justifications for α in O is sufficient for identifying all modules for α in O .”

Modules

- Let O be an ontology and $O_1 \subseteq O$ is a (possibly empty) subset of axioms in O . O_1 is a module for an axiom α in O (an α -module) if: $O_1 \models \alpha$ if and only if $O \models \alpha$.

If an ontology entails an axiom, then a module for the axiom must contain enough for at least one proof of that axiom. A module for a signature contains enough information to decide whether the ontology entails an axiom constructed from the signature. “Hence, knowing all the justifications for α in O is sufficient for identifying all modules for α in O .”

Modules

- Let O be an ontology and $O_1 \subseteq O$ is a (possibly empty) subset of axioms in O . O_1 is a module for an axiom α in O (an α -module) if: $O_1 \models \alpha$ if and only if $O \models \alpha$.
- O_1 is a module for signature \mathbf{S} if for every axiom α whose signature is a subset of \mathbf{S} , O_1 is a module for α .

If an ontology entails an axiom, then a module for the axiom must contain enough for at least one proof of that axiom. A module for a signature contains enough information to decide whether the ontology entails an axiom constructed from the signature. “Hence, knowing all the justifications for α in O is sufficient for identifying all modules for α in O .”

Consider...

- Suppose that the concepts *GeneticFibrosis* and *GeneticOrigin* are empty.
- Can the truth of the following subsumption relation be established?
- $GeneticFibrosis \equiv Fibrosis \sqcap \exists hasOrigin.GeneticOrigin$

GeneticFibrosis \equiv *Fibrosis* \sqcap \exists *hasOrigin.GeneticOrigin*

A definition is simply shorthand for two subsumption propositions. The first is trivially satisfied since it makes a claim that an empty class is subsumed by another. The second is satisfied through several steps.

GeneticFibrosis \equiv *Fibrosis* \sqcap \exists *hasOrigin.GeneticOrigin*



GeneticFibrosis \sqsubseteq *Fibrosis* \sqcap \exists *hasOrigin.GeneticOrigin*

GeneticFibrosis \sqsupseteq *Fibrosis* \sqcap \exists *hasOrigin.GeneticOrigin*

A definition is simply shorthand for two subsumption propositions. The first is trivially satisfied since it makes a claim that an empty class is subsumed by another. The second is satisfied through several steps.

$GeneticFibrosis \equiv Fibrosis \sqcap \exists hasOrigin.GeneticOrigin$



$GeneticFibrosis \sqsubseteq Fibrosis \sqcap \exists hasOrigin.GeneticOrigin$

GeneticFibrosis is empty, so is
trivially a sub-concept of anything.

$GeneticFibrosis \sqsupseteq Fibrosis \sqcap \exists hasOrigin.GeneticOrigin$

A definition is simply shorthand for two subsumption propositions. The first is trivially satisfied since it makes a claim that an empty class is subsumed by another. The second is satisfied through several steps.

$GeneticFibrosis \equiv Fibrosis \sqcap \exists hasOrigin.GeneticOrigin$



$GeneticFibrosis \sqsubseteq Fibrosis \sqcap \exists hasOrigin.GeneticOrigin$

GeneticFibrosis is empty, so is trivially a sub-concept of anything.

$GeneticFibrosis \sqsupseteq Fibrosis \sqcap \exists hasOrigin.GeneticOrigin$

GeneticOrigin is empty...

A definition is simply shorthand for two subsumption propositions. The first is trivially satisfied since it makes a claim that an empty class is subsumed by another. The second is satisfied through several steps.

$GeneticFibrosis \equiv Fibrosis \sqcap \exists hasOrigin.GeneticOrigin$



$GeneticFibrosis \sqsubseteq Fibrosis \sqcap \exists hasOrigin.GeneticOrigin$

GeneticFibrosis is empty, so is trivially a sub-concept of anything.

$GeneticFibrosis \sqsupseteq Fibrosis \sqcap \exists hasOrigin.GeneticOrigin$

GeneticOrigin is empty...

...so nothing *has* such an origin...

A definition is simply shorthand for two subsumption propositions. The first is trivially satisfied since it makes a claim that an empty class is subsumed by another. The second is satisfied through several steps.

$GeneticFibrosis \equiv Fibrosis \sqcap \exists hasOrigin.GeneticOrigin$



$GeneticFibrosis \sqsubseteq Fibrosis \sqcap \exists hasOrigin.GeneticOrigin$

GeneticFibrosis is empty, so is trivially a sub-concept of anything.

$GeneticFibrosis \sqsubseteq Fibrosis \sqcap \exists hasOrigin.GeneticOrigin$

GeneticOrigin is empty...

...so nothing *has* such an origin...

...and so the intersection is empty, and the whole right side is trivially a sub-concept of anything.

A definition is simply shorthand for two subsumption propositions. The first is trivially satisfied since it makes a claim that an empty class is subsumed by another. The second is satisfied through several steps.

$GeneticFibrosis \equiv Fibrosis \sqcap \exists hasOrigin.GeneticOrigin$



$GeneticFibrosis \sqsubseteq Fibrosis \sqcap \exists hasOrigin.GeneticOrigin$

GeneticFibrosis is empty, so is trivially a sub-concept of anything.

$GeneticFibrosis \sqsubseteq Fibrosis \sqcap \exists hasOrigin.GeneticOrigin$

GeneticOrigin is empty...

...so nothing *has* such an origin...

...and so the intersection is empty, and the whole right side is trivially a sub-concept of anything.

Then, provided that *GeneticFibrosis* and *GeneticOrigin* are empty, the definition is true, regardless of the interpretation of *Fibrosis* or *hasOrigin*

A definition is simply shorthand for two subsumption propositions. The first is trivially satisfied since it makes a claim that an empty class is subsumed by another. The second is satisfied through several steps.

Syntactic Locality

- Where \mathbf{S} is a signature, two sets of concepts, $\mathbf{Con}^\emptyset(\mathbf{S})$ and $\mathbf{Con}^\Delta(\mathbf{S})$ are defined:
 - $\mathbf{Con}^\emptyset(\mathbf{S}) ::= A^\emptyset \mid (\neg C^\Delta) \mid (C^\emptyset \sqcap C) \mid (C \sqcap C^\emptyset) \mid (\exists R^\emptyset.C) \mid (\exists R.C^\emptyset) \mid (\geq n R^\emptyset.C) \mid (\geq n R.C^\emptyset)$
 - $\mathbf{Con}^\Delta(\mathbf{S}) ::= (\neg C^\emptyset) \mid (C^\Delta_1 \sqcap C^\Delta_2)$
- Axioms of the form $R^\emptyset \sqsubseteq R$, $\text{Trans}(R^\emptyset)$, $C^\emptyset \sqsubseteq C$, and $C \sqsubseteq C^\Delta$, are *local* to \mathbf{S} .

A^\emptyset refers to any atomic concept not in \mathbf{S} . R^\emptyset is any role (or inverse of a role) not in \mathbf{S} . The empty set superscript indicates a concept which will be interpreted as the empty set, though $\mathbf{Con}^\emptyset(\mathbf{S})$ includes the complements of non-empty sets which may be non-empty. An intersection with an empty set will, of course, be empty, and a role or number restriction where the role or concept are empty must also be empty. The complement of an empty concept must be the entire universe, and the intersection of two non-empty concepts may, or may not, be empty. Importantly, axioms which are local to \mathbf{S} are guaranteed to be true when the elements of $\mathbf{Con}^\emptyset(\mathbf{S})$ are interpreted as the empty set. An empty role or concept is guaranteed to be subsumed by any other role or concept, and any C^Δ must be the entire universe and so subsumes any concept

Algorithm 1

Input:

O : ontology

S : signature

Output:

O_1 : a module for S in O

1. $O_1 \leftarrow \emptyset. O_2 \leftarrow O.$
2. **until** O_2 is empty
3. Remove an axiom α from O_2 .
4. **when** α is local to $S \cup \text{Sig}(O_1)$
7. $O_1 \leftarrow O_1 \cup \{\alpha\}. O_2 \leftarrow O \setminus O_1.$
- || **return** O_1

Note that the α selected on a given iteration will never be present in O^2 at the end of the iteration. However, the condition update to O^2 is that O^2 becomes O minus the contents of O^1 . This means that an α removed on one iteration, but not added to O^1 may appear in O^2 again later (since that α will still be contained in $O \setminus O^1$). The authors prove in another paper that this algorithm generates a module for the signature S , and that it runs in polynomial time of the size of the input ontology O .

Special Result


- **Proposition 2.**

Let O be a *SHOIQ* ontology, $X, Y \in CN(O) \cup \{\top\} \cup \{\perp\}$, and O_X the output of Algorithm 1 for input O and $S = \text{Sig}(X)$.

Then O_X is a module in O for $\alpha = (X \sqsubseteq Y)$.

Propositional Example

KB ¹	
1	P
2	$P \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$




KB ²	
1	P
2	$R \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$

This example is designed to demonstrate that modules can be used to answer questions about entailment, and the only computation necessary is checking whether subset relations hold. There is no need to recompute entire proofs. $\{R \wedge S\}$ entails R , and even though KB^2 is different to KB^1 , it must also entail R as it contains $\{R \wedge S\}$.

Propositional Example

KB ¹	
1	P
2	$P \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$




KB ²	
1	P
2	$R \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$

KB¹ $\models R$...

This example is designed to demonstrate that modules can be used to answer questions about entailment, and the only computation necessary is checking whether subset relations hold. There is no need to recompute entire proofs. $\{R \wedge S\}$ entails R , and even though KB² is different to KB¹, it must also entail R as it contains $\{R \wedge S\}$.

Propositional Example

KB ¹	
1	P
2	$P \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$



KB ²	
1	P
2	$R \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$


KB¹ $\models R$...

... $\{R \wedge S\}$ is a module for R in KB¹...

This example is designed to demonstrate that modules can be used to answer questions about entailment, and the only computation necessary is checking whether subset relations hold. There is no need to recompute entire proofs. $\{R \wedge S\}$ entails R , and even though KB² is different to KB¹, it must also entail R as it contains $\{R \wedge S\}$.

Propositional Example

KB ¹	
1	P
2	$P \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$



KB ²	
1	P
2	$R \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$

KB¹ $\models R$...


... $\{R \wedge S\}$ is a module for R in KB¹...

...and $\{R \wedge S\} \subseteq \text{KB}^2$...

This example is designed to demonstrate that modules can be used to answer questions about entailment, and the only computation necessary is checking whether subset relations hold. There is no need to recompute entire proofs. $\{R \wedge S\}$ entails R , and even though KB² is different to KB¹, it must also entail R as it contains $\{R \wedge S\}$.

Propositional Example

KB ¹	
1	P
2	$P \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$



KB ²	
1	P
2	$R \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$

KB¹ $\models R$...

... $\{R \wedge S\}$ is a module for R in KB¹...


...and $\{R \wedge S\} \subseteq \text{KB}^2$...

...so KB² $\models R$.

This example is designed to demonstrate that modules can be used to answer questions about entailment, and the only computation necessary is checking whether subset relations hold. There is no need to recompute entire proofs. $\{R \wedge S\}$ entails R , and even though KB² is different to KB¹, it must also entail R as it contains $\{R \wedge S\}$.

Propositional Example

KB ¹	
1	P
2	$P \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$




KB ²	
1	P
2	$R \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$

This example is designed to demonstrate that modules can be used to answer questions about entailment, and the only computation necessary is checking whether subset relations hold. There is no need to recompute entire proofs. This case is slightly more interesting. Without knowing whether KB^2 entails W , we can still compute a module for W . We know that KB^1 does not entail W , but yet it contains a subset which is a module for W in O^2 . Then KB^2 cannot entail W either. (As an explanation. $KB^1 \not\models W$. O^2_W is a module for W in O^2 , and thus entails W if and only if O^2 does. If it is a subset of O^1 , however, then it must not entail W since O^1 as a whole does not even entail W . Then neither O^2 entail W .) The important thing to note here is that while we can check very quickly whether KB^2 entails W , and so decide whether $\{P, Q \Rightarrow R\}$ is a module, the module **could** have been computed using a module extraction

Propositional Example

KB ¹	
1	P
2	$P \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$




KB ²	
1	P
2	$R \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$

KB¹ $\not\models$ W...

This example is designed to demonstrate that modules can be used to answer questions about entailment, and the only computation necessary is checking whether subset relations hold. There is no need to recompute entire proofs. This case is slightly more interesting. Without knowing whether KB² entails W, we can still compute a module for W. We know that KB¹ does not entail W, but yet it contains a subset which is a module for W in O². Then KB² cannot entail W either. (As an explanation. KB¹ $\not\models$ W. O²_W is a module for W in O², and thus entails W if and only if O² does. If it is a subset of O¹, however, then it must not entail W since O¹ as a whole does not even entail W. Then neither O² entail W.) The important thing to note here is that while we can check very quickly whether KB² entails W, and so decide whether {P, Q \Rightarrow R} is a module, the module **could** have been computed using a module extraction

Propositional Example

KB ¹	
1	P
2	$P \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$



KB ²	
1	P
2	$R \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$


KB¹ $\not\models$ W...

...{P, Q \Rightarrow R} is a module for W in KB²...

This example is designed to demonstrate that modules can be used to answer questions about entailment, and the only computation necessary is checking whether subset relations hold. There is no need to recompute entire proofs. This case is slightly more interesting. Without knowing whether KB² entails W, we can still compute a module for W. We know that KB¹ does not entail W, but yet it contains a subset which is a module for W in O². Then KB² cannot entail W either. (As an explanation. KB¹ $\not\models$ W. O²_W is a module for W in O², and thus entails W if and only if O² does. If it is a subset of O¹, however, then it must not entail W since O¹ as a whole does not even entail W. Then neither O² entail W.) The important thing to note here is that while we can check very quickly whether KB² entails W, and so decide whether {P, Q \Rightarrow R} is a module, the module **could** have been computed using a module extraction

Propositional Example

KB ¹	
1	P
2	$P \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$



KB ²	
1	P
2	$R \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$

KB¹ $\not\models$ W...


...{P, Q \Rightarrow R} is a module for W in KB²...

...and {P, Q \Rightarrow R} \subseteq KB¹...

This example is designed to demonstrate that modules can be used to answer questions about entailment, and the only computation necessary is checking whether subset relations hold. There is no need to recompute entire proofs. This case is slightly more interesting. Without knowing whether KB² entails W, we can still compute a module for W. We know that KB¹ does not entail W, but yet it contains a subset which is a module for W in O². Then KB² cannot entail W either. (As an explanation. KB¹ $\not\models$ W. O²_W is a module for W in O², and thus entails W if and only if O² does. If it is a subset of O¹, however, then it must not entail W since O¹ as a whole does not even entail W. Then neither O² entail W.) The important thing to note here is that while we can check very quickly whether KB² entails W, and so decide whether {P, Q \Rightarrow R} is a module, the module **could** have been computed using a module extraction

Propositional Example

KB ¹	
1	P
2	$P \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$



KB ²	
1	P
2	$R \Rightarrow Q$
3	$Q \Rightarrow R$
4	$R \wedge S$

KB¹ $\not\models$ W...

...{ $P, Q \Rightarrow R$ } is a module for W in KB²...

...and { $P, Q \Rightarrow R$ } \subseteq KB¹...

...so KB² $\not\models$ W.

This example is designed to demonstrate that modules can be used to answer questions about entailment, and the only computation necessary is checking whether subset relations hold. There is no need to recompute entire proofs. This case is slightly more interesting. Without knowing whether KB² entails W, we can still compute a module for W. We know that KB¹ does not entail W, but yet it contains a subset which is a module for W in O². Then KB² cannot entail W either. (As an explanation. KB¹ $\not\models$ W. O²_W is a module for W in O², and thus entails W if and only if O² does. If it is a subset of O¹, however, then it must not entail W since O¹ as a whole does not even entail W. Then neither O² entail W.) The important thing to note here is that while we can check very quickly whether KB² entails W, and so decide whether { $P, Q \Rightarrow R$ } is a module, the module **could** have been computed using a module extraction

Generalization

- **Proposition 3.**

Let O^1 and O^2 be ontologies, α an axiom, and O^1_α, O^2_α respectively modules for α in O^1 and O^2 . Then:

1. If $O^1 \models \alpha$ and $O^1_\alpha \subseteq O^2$, then $O^2 \models \alpha$.
2. If $O^1 \not\models \alpha$ and $O^2_\alpha \subseteq O^1$, then $O^2 \not\models \alpha$.

Classifying Ontologies

- *Classification* is the process of computing the subsumption relation \sqsubseteq for an ontology.
- The results presented so far can be used to efficiently classify a later version of an ontology produced by an incremental change to an earlier version which has already been classified, and for which modules have already been computed.

Algorithm 2

- The incremental classification algorithm can be separated into three phases:

Algorithm 2 $\text{inc_classify}(\mathcal{O}^1, \Delta\mathcal{O}, \sim_1, X \rightarrow \mathcal{O}_X^1)$

Input:

\mathcal{O}^1 : an ontology

$\Delta\mathcal{O} = (\Delta^-\mathcal{O}, \Delta^+\mathcal{O})$: removed / added axioms

\sqsubseteq_1 : subsumption relations in \mathcal{O}^1

$X \rightarrow \mathcal{O}_X^1$: a module for every $X \in \text{CN}(\mathcal{O}^1) \cup \{\top\}$

Output:

\mathcal{O}^2 : the result of applying the change $\Delta\mathcal{O}$ to \mathcal{O}^1

\sqsubseteq_2 : subsumption relations in \mathcal{O}^2

$X \rightarrow \mathcal{O}_X^2$: a module for every $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$

```
1:  $\mathcal{O}^2 \leftarrow (\mathcal{O}^1 \setminus \Delta^-\mathcal{O}) \cup \Delta^+\mathcal{O}$ 
2: for each  $A \in \text{CN}(\mathcal{O}^2) \setminus \text{CN}(\mathcal{O}^1)$  do
3:    $\mathcal{O}_A^1 \leftarrow \mathcal{O}_\top^1$ 
4:   for each  $\top \sqsubseteq_1 Y$  do  $A \sqsubseteq_1 Y \leftarrow \text{true}$ 
5:   for each  $X \sqsubseteq_1 \perp$  do  $X \sqsubseteq_1 A \leftarrow \text{true}$ 
6: end for
7:  $M^- \leftarrow \emptyset$   $M^+ \leftarrow \emptyset$ 
8: for each  $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$  do
9:   for each  $\alpha \in \Delta^-\mathcal{O}$  do
10:    if not  $s\_local(\alpha, \text{Sig}(\mathcal{O}_X^1))$  then
11:       $M^- \leftarrow M^- \cup \{X\}$ 
12:    end if
13:  end for
14:  for each  $\alpha \in \Delta^+\mathcal{O}$  do
15:    if not  $s\_local(\alpha, \text{Sig}(\mathcal{O}_X^1))$  then
16:       $M^+ \leftarrow M^+ \cup \{X\}$ 
17:    end if
18:  end for
19: end for
20: for each  $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$  do
21:  if  $X \in M^- \cup M^+$  then
22:     $\mathcal{O}_X^2 \leftarrow \text{extract\_module}(\text{Sig}(X), \mathcal{O}^2)$ 
23:  else
24:     $\mathcal{O}_X^2 \leftarrow \mathcal{O}_X^1$ 
25:  end if
26:  for each  $Y \in \text{CN}(\mathcal{O}^2) \cup \{\perp\}$  do
27:    if  $(X \in M^- \text{ and } X \sqsubseteq_1 Y)$  or
28:       $(X \in M^+ \text{ and } X \not\sqsubseteq_1 Y)$  then
29:       $X \sqsubseteq_2 Y \leftarrow \text{test}(\mathcal{O}_X^2 \models X \sqsubseteq Y)$ 
30:    else
31:       $X \sqsubseteq_2 Y \leftarrow X \sqsubseteq_1 Y$ 
32:    end if
33:  end for
34: end for
35: return  $\mathcal{O}^2, \sqsubseteq_2, X \rightarrow \mathcal{O}_X^2$ 
```

Algorithm 2 $\text{inc_classify}(\mathcal{O}^1, \Delta\mathcal{O}, \sim_1, X \rightarrow \mathcal{O}_X^1)$

Input:

\mathcal{O}^1 : an ontology

$\Delta\mathcal{O} = (\Delta^-\mathcal{O}, \Delta^+\mathcal{O})$: removed / added axioms

\sqsubseteq_1 : subsumption relations in \mathcal{O}^1

$X \rightarrow \mathcal{O}_X^1$: a module for every $X \in \text{CN}(\mathcal{O}^1) \cup \{\top\}$

Output:

\mathcal{O}^2 : the result of applying the change $\Delta\mathcal{O}$ to \mathcal{O}^1

\sqsubseteq_2 : subsumption relations in \mathcal{O}^2

$X \rightarrow \mathcal{O}_X^2$: a module for every $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$

Algorithm 2

- The incremental classification algorithm can be separated into three phases:

I. Update “easy” stuff for \mathcal{O}^1 .

```
1:  $\mathcal{O}^2 \leftarrow (\mathcal{O}^1 \setminus \Delta^-\mathcal{O}) \cup \Delta^+\mathcal{O}$ 
2: for each  $A \in \text{CN}(\mathcal{O}^2) \setminus \text{CN}(\mathcal{O}^1)$  do
3:    $\mathcal{O}_A^1 \leftarrow \mathcal{O}_\top^1$ 
4:   for each  $\top \sqsubseteq_1 Y$  do  $A \sqsubseteq_1 Y \leftarrow \text{true}$ 
5:   for each  $X \sqsubseteq_1 \perp$  do  $X \sqsubseteq_1 A \leftarrow \text{true}$ 
6: end for
7:  $M^- \leftarrow \emptyset$   $M^+ \leftarrow \emptyset$ 
8: for each  $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$  do
9:   for each  $\alpha \in \Delta^-\mathcal{O}$  do
10:    if not  $\text{s\_local}(\alpha, \text{Sig}(\mathcal{O}_X^1))$  then
11:       $M^- \leftarrow M^- \cup \{X\}$ 
12:    end if
13:  end for
14:  for each  $\alpha \in \Delta^+\mathcal{O}$  do
15:    if not  $\text{s\_local}(\alpha, \text{Sig}(\mathcal{O}_X^1))$  then
16:       $M^+ \leftarrow M^+ \cup \{X\}$ 
17:    end if
18:  end for
19: end for
20: for each  $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$  do
21:  if  $X \in M^- \cup M^+$  then
22:     $\mathcal{O}_X^2 \leftarrow \text{extract\_module}(\text{Sig}(X), \mathcal{O}^2)$ 
23:  else
24:     $\mathcal{O}_X^2 \leftarrow \mathcal{O}_X^1$ 
25:  end if
26:  for each  $Y \in \text{CN}(\mathcal{O}^2) \cup \{\perp\}$  do
27:    if  $(X \in M^- \text{ and } X \sqsubseteq_1 Y)$  or
28:       $(X \in M^+ \text{ and } X \not\sqsubseteq_1 Y)$  then
29:       $X \sqsubseteq_2 Y \leftarrow \text{test}(\mathcal{O}_X^2 \models X \sqsubseteq Y)$ 
30:    else
31:       $X \sqsubseteq_2 Y \leftarrow X \sqsubseteq_1 Y$ 
32:    end if
33:  end for
34: end for
35: return  $\mathcal{O}^2, \sqsubseteq_2, X \rightarrow \mathcal{O}_X^2$ 
```

Algorithm 2 $\text{inc_classify}(\mathcal{O}^1, \Delta\mathcal{O}, \sim_1, X \rightarrow \mathcal{O}_X^1)$

Input:

\mathcal{O}^1 : an ontology

$\Delta\mathcal{O} = (\Delta^-\mathcal{O}, \Delta^+\mathcal{O})$: removed / added axioms

\sqsubseteq_1 : subsumption relations in \mathcal{O}^1

$X \rightarrow \mathcal{O}_X^1$: a module for every $X \in \text{CN}(\mathcal{O}^1) \cup \{\top\}$

Output:

\mathcal{O}^2 : the result of applying the change $\Delta\mathcal{O}$ to \mathcal{O}^1

\sqsubseteq_2 : subsumption relations in \mathcal{O}^2

$X \rightarrow \mathcal{O}_X^2$: a module for every $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$

```
1:  $\mathcal{O}^2 \leftarrow (\mathcal{O}^1 \setminus \Delta^-\mathcal{O}) \cup \Delta^+\mathcal{O}$ 
2: for each  $A \in \text{CN}(\mathcal{O}^2) \setminus \text{CN}(\mathcal{O}^1)$  do
3:    $\mathcal{O}_A^1 \leftarrow \mathcal{O}_\top^1$ 
4:   for each  $\top \sqsubseteq_1 Y$  do  $A \sqsubseteq_1 Y \leftarrow \text{true}$ 
5:   for each  $X \sqsubseteq_1 \perp$  do  $X \sqsubseteq_1 A \leftarrow \text{true}$ 
6: end for
7:  $M^- \leftarrow \emptyset$   $M^+ \leftarrow \emptyset$ 
8: for each  $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$  do
9:   for each  $\alpha \in \Delta^-\mathcal{O}$  do
10:    if not  $\text{s\_local}(\alpha, \text{Sig}(\mathcal{O}_X^1))$  then
11:       $M^- \leftarrow M^- \cup \{X\}$ 
12:    end if
13:  end for
14:  for each  $\alpha \in \Delta^+\mathcal{O}$  do
15:    if not  $\text{s\_local}(\alpha, \text{Sig}(\mathcal{O}_X^1))$  then
16:       $M^+ \leftarrow M^+ \cup \{X\}$ 
17:    end if
18:  end for
19: end for
20: for each  $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$  do
21:   if  $X \in M^- \cup M^+$  then
22:      $\mathcal{O}_X^2 \leftarrow \text{extract\_module}(\text{Sig}(X), \mathcal{O}^2)$ 
23:   else
24:      $\mathcal{O}_X^2 \leftarrow \mathcal{O}_X^1$ 
25:   end if
26:   for each  $Y \in \text{CN}(\mathcal{O}^2) \cup \{\perp\}$  do
27:    if  $(X \in M^- \text{ and } X \sqsubseteq_1 Y)$  or
28:       $(X \in M^+ \text{ and } X \not\sqsubseteq_1 Y)$  then
29:         $X \sqsubseteq_2 Y \leftarrow \text{test}(\mathcal{O}_X^2 \models X \sqsubseteq Y)$ 
30:      else
31:         $X \sqsubseteq_2 Y \leftarrow X \sqsubseteq_1 Y$ 
32:      end if
33:    end for
34: end for
35: return  $\mathcal{O}^2, \sqsubseteq_2, X \rightarrow \mathcal{O}_X^2$ 
```

Algorithm 2

- The incremental classification algorithm can be separated into three phases:
 1. Update “easy” stuff for \mathcal{O}^1 .
 2. Determine altered modules.

Algorithm 2 $\text{inc_classify}(\mathcal{O}^1, \Delta\mathcal{O}, \sim_1, X \rightarrow \mathcal{O}_X^1)$

Input:

\mathcal{O}^1 : an ontology

$\Delta\mathcal{O} = (\Delta^-\mathcal{O}, \Delta^+\mathcal{O})$: removed / added axioms

\sqsubseteq_1 : subsumption relations in \mathcal{O}^1

$X \rightarrow \mathcal{O}_X^1$: a module for every $X \in \text{CN}(\mathcal{O}^1) \cup \{\top\}$

Output:

\mathcal{O}^2 : the result of applying the change $\Delta\mathcal{O}$ to \mathcal{O}^1

\sqsubseteq_2 : subsumption relations in \mathcal{O}^2

$X \rightarrow \mathcal{O}_X^2$: a module for every $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$

```
1:  $\mathcal{O}^2 \leftarrow (\mathcal{O}^1 \setminus \Delta^-\mathcal{O}) \cup \Delta^+\mathcal{O}$ 
2: for each  $A \in \text{CN}(\mathcal{O}^2) \setminus \text{CN}(\mathcal{O}^1)$  do
3:    $\mathcal{O}_A^1 \leftarrow \mathcal{O}_\top^1$ 
4:   for each  $\top \sqsubseteq_1 Y$  do  $A \sqsubseteq_1 Y \leftarrow \text{true}$ 
5:   for each  $X \sqsubseteq_1 \perp$  do  $X \sqsubseteq_1 A \leftarrow \text{true}$ 
6: end for
7:  $M^- \leftarrow \emptyset$   $M^+ \leftarrow \emptyset$ 
8: for each  $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$  do
9:   for each  $\alpha \in \Delta^-\mathcal{O}$  do
10:    if not  $\text{s\_local}(\alpha, \text{Sig}(\mathcal{O}_X^1))$  then
11:       $M^- \leftarrow M^- \cup \{X\}$ 
12:    end if
13:  end for
14:  for each  $\alpha \in \Delta^+\mathcal{O}$  do
15:    if not  $\text{s\_local}(\alpha, \text{Sig}(\mathcal{O}_X^1))$  then
16:       $M^+ \leftarrow M^+ \cup \{X\}$ 
17:    end if
18:  end for
19: end for
20: for each  $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$  do
21:  if  $X \in M^- \cup M^+$  then
22:     $\mathcal{O}_X^2 \leftarrow \text{extract\_module}(\text{Sig}(X), \mathcal{O}^2)$ 
23:  else
24:     $\mathcal{O}_X^2 \leftarrow \mathcal{O}_X^1$ 
25:  end if
26:  for each  $Y \in \text{CN}(\mathcal{O}^2) \cup \{\perp\}$  do
27:    if  $(X \in M^- \text{ and } X \sqsubseteq_1 Y)$  or
28:       $(X \in M^+ \text{ and } X \not\sqsubseteq_1 Y)$  then
29:       $X \sqsubseteq_2 Y \leftarrow \text{test}(\mathcal{O}_X^2 \models X \sqsubseteq Y)$ 
30:    else
31:       $X \sqsubseteq_2 Y \leftarrow X \sqsubseteq_1 Y$ 
32:    end if
33:  end for
34: end for
35: return  $\mathcal{O}^2, \sqsubseteq_2, X \rightarrow \mathcal{O}_X^2$ 
```

Algorithm 2

- The incremental classification algorithm can be separated into three phases:

1. Update “easy” stuff for \mathcal{O}^1 .
2. Determine altered modules.
3. Reclassify concepts whose modules changed in such a way as to require it.

Algorithm 2 $\text{inc_classify}(\mathcal{O}^1, \Delta\mathcal{O}, \sqsubseteq_1, X \mapsto \mathcal{O}_X^1)$

Input:

\mathcal{O}^1 : an ontology
 $\Delta\mathcal{O} = (\Delta^-\mathcal{O}, \Delta^+\mathcal{O})$: removed / added axioms
 \sqsubseteq_1 : subsumption relations in \mathcal{O}^1
 $X \mapsto \mathcal{O}_X^1$: a module for every $X \in \text{CN}(\mathcal{O}^1) \cup \{\top\}$

Output:

\mathcal{O}^2 : the result of applying the change $\Delta\mathcal{O}$ to \mathcal{O}^1
 \sqsubseteq_2 : subsumption relations in \mathcal{O}^2
 $X \mapsto \mathcal{O}_X^2$: a module for every $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$

```
1:  $\mathcal{O}^2 \leftarrow (\mathcal{O}^1 \setminus \Delta^-\mathcal{O}) \cup \Delta^+\mathcal{O}$ 
2: for each  $A \in \text{CN}(\mathcal{O}^2) \setminus \text{CN}(\mathcal{O}^1)$  do
3:    $\mathcal{O}_A^1 \leftarrow \mathcal{O}_\top^1$ 
4:   for each  $\top \sqsubseteq_1 Y$  do  $A \sqsubseteq_1 Y \leftarrow \text{true}$ 
5:   for each  $X \sqsubseteq_1 \perp$  do  $X \sqsubseteq_1 A \leftarrow \text{true}$ 
6: end for
7:  $M^- \leftarrow \emptyset$   $M^+ \leftarrow \emptyset$ 
8: for each  $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$  do
9:   for each  $\alpha \in \Delta^-\mathcal{O}$  do
10:    if not  $\text{s\_local}(\alpha, \text{Sig}(\mathcal{O}_X^1))$  then
11:       $M^- \leftarrow M^- \cup \{X\}$ 
12:    end if
13:  end for
14:  for each  $\alpha \in \Delta^+\mathcal{O}$  do
15:    if not  $\text{s\_local}(\alpha, \text{Sig}(\mathcal{O}_X^1))$  then
16:       $M^+ \leftarrow M^+ \cup \{X\}$ 
17:    end if
18:  end for
19: end for
20: for each  $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$  do
21:   if  $X \in M^- \cup M^+$  then
22:      $\mathcal{O}_X^2 \leftarrow \text{extract\_module}(\text{Sig}(X), \mathcal{O}^2)$ 
23:   else
24:      $\mathcal{O}_X^2 \leftarrow \mathcal{O}_X^1$ 
25:   end if
26:   for each  $Y \in \text{CN}(\mathcal{O}^2) \cup \{\perp\}$  do
27:    if  $(X \in M^- \text{ and } X \sqsubseteq_1 Y)$  or
28:     $(X \in M^+ \text{ and } X \not\sqsubseteq_1 Y)$  then
29:       $X \sqsubseteq_2 Y \leftarrow \text{test}(\mathcal{O}_X^2 \models X \sqsubseteq Y)$ 
30:    else
31:       $X \sqsubseteq_2 Y \leftarrow X \sqsubseteq_1 Y$ 
32:    end if
33:   end for
34: end for
35: return  $\mathcal{O}^2, \sqsubseteq_2, X \mapsto \mathcal{O}_X^2$ 
```

Algorithm 2 (Input/Output)

Input:

\mathcal{O}^1 : an ontology
 $\Delta\mathcal{O} = (\Delta^-\mathcal{O}, \Delta^+\mathcal{O})$: removed/added axioms
 \sqsubseteq_1 : subsumption relation for \mathcal{O}^1

Output:

\mathcal{O}^2 : the result of applying the change $\Delta\mathcal{O}$ to \mathcal{O}^1
 \sqsubseteq_2 : subsumption relations in \mathcal{O}^2
 $X \mapsto \mathcal{O}_X^2$: a module for every $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$

1. $\mathcal{O}^2 \leftarrow (\mathcal{O}^1 \setminus \Delta^-\mathcal{O}) \cup \Delta^+\mathcal{O}$

As input, **Algorithm 2** requires the previous ontology, the incremental changes (both additions and removals), and the subsumption relation for the previous ontology. (Note that the changes are not particularly sophisticated. E.g., **Algorithm 2** can't make use of the fact that an intersection in a definition simply added another argument. The older axioms is removed, and the new version is added.)

The output of **Algorithm 2** is the new ontology \mathcal{O}^2 (which is trivial to construct, so this isn't a particularly interesting result, see step 1), the subsumption relation for \mathcal{O}^2 , and a function mapping concept names to modules for those names in \mathcal{O}^2 .

Algorithm 2 (lines 2–6)

Algorithm 2 $\text{inc_classify}(\mathcal{O}^1, \Delta\mathcal{O}, \sim_1, X, \mathcal{O}_X^1)$

Input:

\mathcal{O}^1 : an ontology
 $\Delta\mathcal{O} = (\Delta^-\mathcal{O}, \Delta^+\mathcal{O})$: removed / added axioms
 \sqsubseteq_1 : subsumption relations in \mathcal{O}^1
 $X \rightarrow \mathcal{O}_X^1$: a module for every $X \in \text{CN}(\mathcal{O}^1) \cup \{\top\}$

Output:

\mathcal{O}^2 : the result of applying the change $\Delta\mathcal{O}$ to \mathcal{O}^1
 \sqsubseteq_2 : subsumption relations in \mathcal{O}^2
 $X \rightarrow \mathcal{O}_X^2$: a module for every $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$

```

1:  $\mathcal{O}^2 \leftarrow (\mathcal{O}^1 \setminus \Delta^-\mathcal{O}) \cup \Delta^+\mathcal{O}$ 
2: for each  $A \in \text{CN}(\mathcal{O}^2) \setminus \text{CN}(\mathcal{O}^1)$  do
3:    $\mathcal{O}_A^1 \leftarrow \mathcal{O}_\top^1$ 
4:   for each  $\top \sqsubseteq_1 Y$  do  $A \sqsubseteq_1 Y \leftarrow \text{true}$ 
5:   for each  $X \sqsubseteq_1 \perp$  do  $X \sqsubseteq_1 A \leftarrow \text{true}$ 
6: end for
7:  $M^- \leftarrow \emptyset$   $M^+ \leftarrow \emptyset$ 
8: for each  $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$  do
9:   for each  $\alpha \in \Delta^-\mathcal{O}$  do
10:    if not  $s\_local(\alpha, \text{Sig}(\mathcal{O}_X^1))$  then
11:       $M^- \leftarrow M^- \cup \{X\}$ 
12:    end if
13:  end for
14:  for each  $\alpha \in \Delta^+\mathcal{O}$  do
15:    if not  $s\_local(\alpha, \text{Sig}(\mathcal{O}_X^1))$  then
16:       $M^+ \leftarrow M^+ \cup \{X\}$ 
17:    end if
18:  end for
19: end for
20: for each  $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$  do
21:  if  $X \in M^- \cup M^+$  then
22:     $\mathcal{O}_X^2 \leftarrow \text{extract\_module}(\text{Sig}(X), \mathcal{O}^2)$ 
23:  else
24:     $\mathcal{O}_X^2 \leftarrow \mathcal{O}_X^1$ 
25:  end if
26:  for each  $Y \in \text{CN}(\mathcal{O}^2) \cup \{\perp\}$  do
27:    if  $(X \in M^- \text{ and } X \sqsubseteq_1 Y)$  or
28:     $(X \in M^+ \text{ and } X \not\sqsubseteq_1 Y)$  then
29:       $X \sqsubseteq_2 Y \leftarrow \text{test}(\mathcal{O}_X^2 \models X \sqsubseteq Y)$ 
30:    else
31:       $X \sqsubseteq_2 Y \leftarrow X \sqsubseteq_1 Y$ 
32:    end if
33:  end for
34: end for
35: return  $\mathcal{O}^2, \sqsubseteq_2, X \rightarrow \mathcal{O}_X^2$ 

```

- For each new atomic concept A:
 - For any Y such that $\top \sqsubseteq_1 Y$, it must hold that $A \sqsubseteq_1 Y$.
 - For any X such that $X \sqsubseteq_1 \perp$, it must hold that $X \sqsubseteq_1 A$.
 - An appropriate module for A in \mathcal{O}^1 is the module for \top in \mathcal{O}^1 . I.e., \mathcal{O}_\top^1 is a module for A in \mathcal{O}^1 .

These relationships all hold for any new atomic symbol A.

- If the universe is a subset of Y, then Y must include A (no matter what A is).
- If X is a subset in the empty set, then X must be a subset of A (no matter what A is).
- Nothing has been said about A in \mathcal{O}^1 , so the module for A in \mathcal{O}^1 is the same as the module for the empty signature, i.e., \mathcal{O}_\top^1 .

Algorithm 2 $\text{inc_classify}(\mathcal{O}^1, \Delta\mathcal{O}, \sqsubseteq_1, X \rightarrow \mathcal{O}_X^1)$

Input:

\mathcal{O}^1 : an ontology

$\Delta\mathcal{O} = (\Delta^-\mathcal{O}, \Delta^+\mathcal{O})$: removed / added axioms

\sqsubseteq_1 : subsumption relations in \mathcal{O}^1

$X \rightarrow \mathcal{O}_X^1$: a module for every $X \in \text{CN}(\mathcal{O}^1) \cup \{\top\}$

Output:

\mathcal{O}^2 : the result of applying the change $\Delta\mathcal{O}$ to \mathcal{O}^1

\sqsubseteq_2 : subsumption relations in \mathcal{O}^2

$X \rightarrow \mathcal{O}_X^2$: a module for every $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$

```
1:  $\mathcal{O}^2 \leftarrow (\mathcal{O}^1 \setminus \Delta^-\mathcal{O}) \cup \Delta^+\mathcal{O}$ 
2: for each  $A \in \text{CN}(\mathcal{O}^2) \setminus \text{CN}(\mathcal{O}^1)$  do
3:    $\mathcal{O}_A^1 \leftarrow \mathcal{O}_\top^1$ 
4:   for each  $\top \sqsubseteq_1 Y$  do  $A \sqsubseteq_1 Y \leftarrow \text{true}$ 
5:   for each  $X \sqsubseteq_1 \perp$  do  $X \sqsubseteq_1 A \leftarrow \text{true}$ 
6: end for
7:  $M^- \leftarrow \emptyset$   $M^+ \leftarrow \emptyset$ 
8: for each  $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$  do
9:   for each  $\alpha \in \Delta^-\mathcal{O}$  do
10:    if not  $\text{s\_local}(\alpha, \text{Sig}(\mathcal{O}_X^1))$  then
11:       $M^- \leftarrow M^- \cup \{X\}$ 
12:    end if
13:  end for
14:  for each  $\alpha \in \Delta^+\mathcal{O}$  do
15:    if not  $\text{s\_local}(\alpha, \text{Sig}(\mathcal{O}_X^1))$  then
16:       $M^+ \leftarrow M^+ \cup \{X\}$ 
17:    end if
18:  end for
19: end for
20: for each  $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$  do
21:   if  $X \in M^- \cup M^+$  then
22:      $\mathcal{O}_X^2 \leftarrow \text{extract\_module}(\text{Sig}(X), \mathcal{O}^2)$ 
23:   else
24:      $\mathcal{O}_X^2 \leftarrow \mathcal{O}_X^1$ 
25:   end if
26:   for each  $Y \in \text{CN}(\mathcal{O}^2) \cup \{\perp\}$  do
27:    if  $(X \in M^- \text{ and } X \sqsubseteq_1 Y)$  or
28:     $(X \in M^+ \text{ and } X \not\sqsubseteq_1 Y)$  then
29:       $X \sqsubseteq_2 Y \leftarrow \text{test}(\mathcal{O}_X^2 \models X \sqsubseteq Y)$ 
30:    else
31:       $X \sqsubseteq_2 Y \leftarrow X \sqsubseteq_1 Y$ 
32:    end if
33:  end for
34: end for
35: return  $\mathcal{O}^2, \sqsubseteq_2, X \rightarrow \mathcal{O}_X^2$ 
```

Algorithm 2 (lines 7–19)

- For each concept X , if an axiom α is non-local with respect to $\text{Sig}(\mathcal{O}_X^1)$, and is
 - *added*: α should be added to \mathcal{O}_X^1 .
 - *removed*: α should be removed from \mathcal{O}_X^1 .

Recall that axioms that are local to a signature are those which are guaranteed to be **true**. A **non-local** axiom is not guaranteed to be true (but may be true or false). \mathcal{O}_X^1 is the module for X in \mathcal{O}^1 and so it entails X if and only if \mathcal{O}^1 entails X . Thus, if an axiom is non-local to the signature of \mathcal{O}_X^1 , then it may have some effect on whether X is entailed by the module. Thus if it is added to the ontology, it should be added to the module, and if removed, removed.

Algorithm 2 $\text{inc_classify}(\mathcal{O}^1, \Delta\mathcal{O}, \sqsubseteq_1, X \rightarrow \mathcal{O}_X^1)$

Input:

\mathcal{O}^1 : an ontology
 $\Delta\mathcal{O} = (\Delta^-\mathcal{O}, \Delta^+\mathcal{O})$: removed / added axioms
 \sqsubseteq_1 : subsumption relations in \mathcal{O}^1
 $X \rightarrow \mathcal{O}_X^1$: a module for every $X \in \text{CN}(\mathcal{O}^1) \cup \{\top\}$

Output:

\mathcal{O}^2 : the result of applying the change $\Delta\mathcal{O}$ to \mathcal{O}^1
 \sqsubseteq_2 : subsumption relations in \mathcal{O}^2
 $X \rightarrow \mathcal{O}_X^2$: a module for every $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$

```
1:  $\mathcal{O}^2 \leftarrow (\mathcal{O}^1 \setminus \Delta^-\mathcal{O}) \cup \Delta^+\mathcal{O}$ 
2: for each  $A \in \text{CN}(\mathcal{O}^2) \setminus \text{CN}(\mathcal{O}^1)$  do
3:    $\mathcal{O}_A^1 \leftarrow \mathcal{O}_\top^1$ 
4:   for each  $\top \sqsubseteq_1 Y$  do  $A \sqsubseteq_1 Y \leftarrow \text{true}$ 
5:   for each  $X \sqsubseteq_1 \perp$  do  $X \sqsubseteq_1 A \leftarrow \text{true}$ 
6: end for
7:  $M^- \leftarrow \emptyset$   $M^+ \leftarrow \emptyset$ 
8: for each  $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$  do
9:   for each  $\alpha \in \Delta^-\mathcal{O}$  do
10:    if not  $\text{s\_local}(\alpha, \text{Sig}(\mathcal{O}_X^1))$  then
11:       $M^- \leftarrow M^- \cup \{X\}$ 
12:    end if
13:  end for
14:  for each  $\alpha \in \Delta^+\mathcal{O}$  do
15:    if not  $\text{s\_local}(\alpha, \text{Sig}(\mathcal{O}_X^1))$  then
16:       $M^+ \leftarrow M^+ \cup \{X\}$ 
17:    end if
18:  end for
19: end for
20: for each  $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$  do
21:   if  $X \in M^- \cup M^+$  then
22:      $\mathcal{O}_X^2 \leftarrow \text{extract\_module}(\text{Sig}(X), \mathcal{O}^2)$ 
23:   else
24:      $\mathcal{O}_X^2 \leftarrow \mathcal{O}_X^1$ 
25:   end if
26:   for each  $Y \in \text{CN}(\mathcal{O}^2) \cup \{\perp\}$  do
27:    if  $(X \in M^- \text{ and } X \sqsubseteq_1 Y)$  or
28:     $(X \in M^+ \text{ and } X \not\sqsubseteq_1 Y)$  then
29:       $X \sqsubseteq_2 Y \leftarrow \text{test}(\mathcal{O}_X^2 \models X \sqsubseteq Y)$ 
30:    else
31:       $X \sqsubseteq_2 Y \leftarrow X \sqsubseteq_1 Y$ 
32:    end if
33:  end for
34: end for
35: return  $\mathcal{O}^2, \sqsubseteq_2, X \rightarrow \mathcal{O}_X^2$ 
```

Algorithm 2 (lines 20–34)

- M^- and M^+ now contain all the concepts whose modules must be recomputed. Modules for other concepts are unchanged.
- Then, for all concepts X and Y in \mathcal{O}^2 , if $[X \sqsubseteq_1 Y \text{ and } X \in M^-]$, or $[X \not\sqsubseteq_1 Y \text{ and } X \in M^+]$, then $X \sqsubseteq_2 Y$ must be computed, but can be checked with \mathcal{O}_X^2 rather than the entire \mathcal{O}^2 . Otherwise, $X \sqsubseteq_2 Y$ if and only if $X \sqsubseteq_1 Y$.

(I don't think that this should really read $Y \sqsubseteq_1 X$, but rather $\mathcal{O}^1 \models Y \sqsubseteq_1 X$.) M^- and M^+ are now all the concepts whose modules must be recomputed. If Y previously subsumed X , and axioms have been removed from the module for X , or Y didn't subsume X , but axioms have been added to the module for X , then whether Y subsumes X must be recomputed. Even here, however, the check can involve just \mathcal{O}_X^2 , which is likely to be substantially smaller than \mathcal{O}^2 . In any other case, the subsumption relationship between a given X and Y is the same between \mathcal{O}^1 and \mathcal{O}^2 .

Module extraction (line 22) is performed by **Algorithm 1** which was explained earlier. The entailment check (line 29) is performed using any off-the-shelf reasoner. Importantly, the

Results

- O^2 has been produced simply by removing Δ^-O from O^1 and adding Δ^+O .
- Ξ_2 has been completely defined.
- For each X , a module O^2_X has been defined.

Evaluation

- Evaluation made use of four ontologies: the National Cancer Institute's *Cancer Ontology*, the *Gene Ontology*, *Generalized Architecture for Languages, Encyclopædias and Nomenclatures in medicine (GALEN)*, and NASA's *SWEET*.

- For each ontology, for various values of n :
 1. Remove n random axioms.
 2. Classify the resulting ontology using Pellet.
- **repeat** 50 times:
 3. Extract the minimal locality-based module for each atomic concept.
 4. Remove an additional n axioms, add the previously removed n axioms, and reclassify.

Algorithm 1 isn't guaranteed to give minimal modules, so it's not clear how in step 3 a "minimal locality-based module" is extracted.

Ontology	Logic	# Concept Names	# Axioms	Class. Time (s.)	% Subs	Init. Mod. Extract (s.)	Mod. Size (Avg/Max)	Non-Loc. Axioms
SWEET	<i>SHOIF</i>	1400	2573	3.6	0.37	1.05	76 / 420	28
Galen	<i>SHF</i>	2749	4529	15.7	0.37	4.8	75 / 530	0
GO	<i>EL</i>	22357	34980	63	0.04	69.6	17.6 / 161	0
NCI	<i>EL</i>	27772	46940	41.1	0.03	76.5	28.9 / 436	0

Table 5. Test suite ontologies.

	<i>n</i>	1: # Mod. Affected (Av / Mx)	2: # Axioms in Aff. Mod. (Av / Mx)	3: Update Aff. Mod. (Av / Mx)	4: Re-class. Aff. Mod. (Av / Mx)	5. Total Time (Av / Mx)	6: # New (Non)Sub. (Av / Mx)	7: # Mod. (Non)Sub (Av / Mx)
NCI	2	67 / 936	545 / 3025	.81 / 5.4	.21 / 1.2	1.03 / 6.77	54 / 1268	17 / 348
SWEET	2	36.9 / 300	281 / 857	.097 / .929	.182 / 1.4	.280 / 2.3	39 / 686	20.1 / 255
Galen	2	134 / 1045	1003 / 2907	.833 / 3.6	2.8 / 13	3.6 / 16.5	111 / 1594	17 / 158
GO	1	39.2 / 1513	127 / 1896	.24 / 1.4	.05 / .47	.29 / 1.5	69 / 2964	33 / 1499
GO	2	46 / 891	216 / 1383	.5 / 2.8	.07 / .43	.57 / 3.2	51 / 1079	26 / 775
GO	4	97 / 1339	474 / 3021	1.4 / 10.1	.25 / 3.2	1.7 / 13.4	94 / 1291	44 / 1034

Table 6. Results for varying update sizes for class and role axioms. Time in seconds.

First the initial module extraction time is highlighted. Then some change is made to the ontology. We can compare the Total time in re-classification with the original classification time. It should be noted that the initial module extraction phase /is/ overhead, or an initial /startup/ cost, but after the first time, the alternatives would be to re-classify using the incremental strategy, or doing a re-classification from scratch. Finally, the three rows of Gene Ontology information suggest (note that the value of *n* is increasing) that the time required for incremental re-classification may grow linearly with the number of axioms modified.

Ontology	Logic	# Concept Names	# Axioms	Class. Time (s.)	% Subs	Init. Mod. Extract (s.)	Mod. Size (Avg/Max)	Non-Loc. Axioms
SWEET	<i>SHOIF</i>	1400	2573	3.6	0.37	1.05	76 / 420	28
Galen	<i>SHF</i>	2749	4529	15.7	0.37	4.8	75 / 530	0
GO	<i>EL</i>	22357	34980	63	0.04	69.6	17.6 / 161	0
NCI	<i>EL</i>	27772	46940	41.1	0.03	76.5	28.9 / 436	0

Table 5. Test suite ontologies.

	<i>n</i>	1: # Mod. Affected (Av / Mx)	2: # Axioms in Aff. Mod. (Av / Mx)	3: Update Aff. Mod. (Av / Mx)	4: Re-class. Aff. Mod. (Av / Mx)	5. Total Time (Av / Mx)	6: # New (Non)Sub. (Av / Mx)	7: # Mod. (Non)Sub (Av / Mx)
NCI	2	67 / 936	545 / 3025	.81 / 5.4	.21 / 1.2	1.03 / 6.77	54 / 1268	17 / 348
SWEET	2	36.9 / 300	281 / 857	.097 / .929	.182 / 1.4	.280 / 2.3	39 / 686	20.1 / 255
Galen	2	134 / 1045	1003 / 2907	.833 / 3.6	2.8 / 13	3.6 / 16.5	111 / 1594	17 / 158
GO	1	39.2 / 1513	127 / 1896	.24 / 1.4	.05 / .47	.29 / 1.5	69 / 2964	33 / 1499
GO	2	46 / 891	216 / 1383	.5 / 2.8	.07 / .43	.57 / 3.2	51 / 1079	26 / 775
GO	4	97 / 1339	474 / 3021	1.4 / 10.1	.25 / 3.2	1.7 / 13.4	94 / 1291	44 / 1034

Table 6. Results for varying update sizes for class and role axioms. Time in seconds.

First the initial module extraction time is highlighted. Then some change is made to the ontology. We can compare the Total time in re-classification with the original classification time. It should be noted that the initial module extraction phase /is/ overhead, or an initial /startup/ cost, but after the first time, the alternatives would be to re-classify using the incremental strategy, or doing a re-classification from scratch. Finally, the three rows of Gene Ontology information suggest (note that the value of *n* is increasing) that the time required for incremental re-classification may grow linearly with the number of axioms modified.

Ontology	Logic	# Concept Names	# Axioms	Class. Time (s.)	% Subs	Init. Mod. Extract (s.)	Mod. Size (Avg/Max)	Non-Loc. Axioms
SWEET	<i>SHOIF</i>	1400	2573	3.6	0.37	1.05	76 / 420	28
Galen	<i>SHF</i>	2749	4529	15.7	0.37	4.8	75 / 530	0
GO	<i>EL</i>	22357	34980	63	0.04	69.6	17.6 / 161	0
NCI	<i>EL</i>	27772	46940	41.1	0.03	76.5	28.9 / 436	0

Table 5. Test suite ontologies.

	n	1: # Mod. Affected (Av / Mx)	2: # Axioms in Aff. Mod. (Av / Mx)	3: Update Aff. Mod. (Av / Mx)	4: Re-class. Aff. Mod. (Av / Mx)	5. Total Time (Av / Mx)	6: # New (Non)Sub. (Av / Mx)	7: # Mod. (Non)Sub (Av / Mx)
NCI	2	67 / 936	545 / 3025	.81 / 5.4	.21 / 1.2	1.03 / 6.77	54 / 1268	17 / 348
SWEET	2	36.9 / 300	281 / 857	.097 / .929	.182 / 1.4	.280 / 2.3	39 / 686	20.1 / 255
Galen	2	134 / 1045	1003 / 2907	.833 / 3.6	2.8 / 13	3.6 / 16.5	111 / 1594	17 / 158
GO	1	39.2 / 1513	127 / 1896	.24 / 1.4	.05 / .47	.29 / 1.5	69 / 2964	33 / 1499
GO	2	46 / 891	216 / 1383	.5 / 2.8	.07 / .43	.57 / 3.2	51 / 1079	26 / 775
GO	4	97 / 1339	474 / 3021	1.4 / 10.1	.25 / 3.2	1.7 / 13.4	94 / 1291	44 / 1034

Table 6. Results for varying update sizes for class and role axioms. Time in seconds.

First the initial module extraction time is highlighted. Then some change is made to the ontology. We can compare the Total time in re-classification with the original classification time. It should be noted that the initial module extraction phase /is/ overhead, or an initial /startup/ cost, but after the first time, the alternatives would be to re-classify using the incremental strategy, or doing a re-classification from scratch. Finally, the three rows of Gene Ontology information suggest (note that the value of n is increasing) that the time required for incremental re-classification may grow linearly with the number of axioms modified.

Ontology	Logic	# Concept Names	# Axioms	Class. Time (s.)	% Subs	Init. Mod. Extract (s.)	Mod. Size (Avg/Max)	Non-Loc. Axioms
SWEET	<i>SHOIF</i>	1400	2573	3.6	0.37	1.05	76 / 420	28
Galen	<i>SHF</i>	2749	4529	15.7	0.37	4.8	75 / 530	0
GO	<i>EL</i>	22357	34980	63	0.04	69.6	17.6 / 161	0
NCI	<i>EL</i>	27772	46940	41.1	0.03	76.5	28.9 / 436	0

Table 5. Test suite ontologies.

	<i>n</i>	1: # Mod. Affected (Av / Mx)	2: # Axioms in Aff. Mod. (Av / Mx)	3: Update Aff. Mod. (Av / Mx)	4: Re-class. Aff. Mod. (Av / Mx)	5. Total Time (Av / Mx)	6: # New (Non)Sub. (Av / Mx)	7: # Mod. (Non)Sub (Av / Mx)
NCI	2	67 / 936	545 / 3025	.81 / 5.4	.21 / 1.2	1.03 / 6.77	54 / 1268	17 / 348
SWEET	2	36.9 / 300	281 / 857	.097 / .929	.182 / 1.4	.280 / 2.3	39 / 686	20.1 / 255
Galen	2	134 / 1045	1003 / 2907	.833 / 3.6	2.8 / 13	3.6 / 16.5	111 / 1594	17 / 158
GO	1	39.2 / 1513	127 / 1896	.24 / 1.4	.05 / .47	.29 / 1.5	69 / 2964	33 / 1499
GO	2	46 / 891	216 / 1383	.5 / 2.8	.07 / .43	.57 / 3.2	51 / 1079	26 / 775
GO	4	97 / 1339	474 / 3021	1.4 / 10.1	.25 / 3.2	1.7 / 13.4	94 / 1291	44 / 1034

Table 6. Results for varying update sizes for class and role axioms. Time in seconds.

First the initial module extraction time is highlighted. Then some change is made to the ontology. We can compare the Total time in re-classification with the original classification time. It should be noted that the initial module extraction phase /is/ overhead, or an initial /startup/ cost, but after the first time, the alternatives would be to re-classify using the incremental strategy, or doing a re-classification from scratch. Finally, the three rows of Gene Ontology information suggest (note that the value of *n* is increasing) that the time required for incremental re-classification may grow linearly with the number of axioms modified.

	<i>n</i>	1: # Mod. Affected (Av / Mx)	2: # Axioms in Aff. Mod. (Av / Mx)	3: Update Aff. Mod. (Av / Mx)	4: Re-class. Aff. Mod. (Av / Mx)	5. Total Time (Av / Mx)	6: # New (Non)Sub. (Av / Mx)	7: # Mod. (Non)Sub (Av / Mx)
NCI	2	2274 / 10217	12161 / 29091	25.7 / 60.4	10.4 / 30.8	36.2 / 91.3	0 / 0	0 / 0
SWEET	2	116 / 296	411 / 956	.42 / .93	.6 / 1.4	1.03 / 2.33	.56 / 28	.28 / 14
Galen	2	524 / 1906	1813 / 3780	2.1 / 4.7	6.5 / 15.6	8.6 / 20.4	3.3 / 82	2.5 / 37

Table 7. Results for varying update sizes for role axiom changes only. Time in seconds.

	<i>n</i>	1: # Mod. Affected (Av / Mx)	2: # Axioms in Aff. Mod. (Av / Mx)	3: Update Aff. Mod. (Av / Mx)	4: Re-class. Aff. Mod. (Av / Mx)	5. Total Time (Av / Mx)	6: # New (Non)Sub. (Av / Mx)	7: # Mod. (Non)Sub (Av / Mx)
NCI	2	33 / 847	396 / 5387	.59 / 8.7	.15 / 2.6	.75 / 11.4	67 / 2228	20 / 610
SWEET	2	15.2 / 243	276 / 800	.02 / .07	.07 / .65	.095 / .732	31 / 553	12 / 241
Galen	2	131 / 1463	913 / 3397	.84 / 4.5	2.6 / 15.4	3.4 / 19.5	69 / 4323	42 / 1178

Table 8. Results for varying update sizes for concept axiom changes only. Time in seconds.

Changes to role axioms seem to require much more work in re-classification than changes to role axioms. The number of modules affected, the size of those modules, and the amount of time needed to re-classify.